Institute of Software Technology
Reliable Software Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Fachstudie

# Evaluating Open-source Tool Stacks for Application Performance Diagnostics

Jan Ruthardt
Nico Poier
Thomas Breunig

| | |
|---|---|
| **Course of Study:** | Softwaretechnik |
| **Examiner:** | Dr.-Ing. André van Hoorn (Prof.-Vertr.) |
| **Supervisor:** | Thomas F. Düllmann M.Sc.,<br>Teerat Pitakrat, M.Sc. |
| **Commenced:** | August 14, 2017 |
| **Completed:** | February 14, 2018 |
| **CR-Classification:** | I.7.2 |

# Abstract

Evaluation of different Open-Source Application Performance Management tools and stacks. Testing of the tools takes places at the RSS Infrastructure (Kubernetes Cluster) and a instance of Sockshop (Microservice Webshop). The goal is to set up a Stack of different tools that is best for monitoring the RSS Infrastructure.

# Contents

1. Introduction vii

2. Technical Data ix
   2.1. General Stack . . . . . ix
   2.2. Collector . . . . . ix
   2.3. Database . . . . . x
   2.4. Visualization . . . . . xi
   2.5. Alerting . . . . . xiii
   2.6. Monitoring enviroment . . . . . xiii

3. Tools xv
   3.1. Template . . . . . xv
   3.2. Installed Tools . . . . . xvi
   3.3. Failed Tools . . . . . xx

4. Conclusion xxiii

A. LaTeX-Tipps xxv
   A.1. File-Encoding und Unterstützung von Umlauten . . . . . xxv
   A.2. Zitate . . . . . xxv
   A.3. Mathematische Formeln . . . . . xxvi
   A.4. Quellcode . . . . . xxvi
   A.5. Abbildungen . . . . . xxvi
   A.6. Tabellen . . . . . xxvii
   A.7. Pseudocode . . . . . xxvii
   A.8. Abkürzungen . . . . . xxxi
   A.9. Verweise . . . . . xxxi
   A.10.Definitionen . . . . . xxxii
   A.11.Verschiedenes . . . . . xxxii
   A.12.Weitere Illustrationen . . . . . xxxii

A.13.Schlusswort . . . . . xxxvi

Bibliography xxxvii

Chapter 1

# Introduction

Nowadays its very Common in IT to have Distributed Systems in Location all over the Globe. To be able to provide the best user Experience its important to Monitor these Networks by only few People sitting in one or more Location. Important here is the Availability and Reliability also as the Response Time of the System.

To tackle these kinds of Tasks Application Performance Management Tool were build. They are available in a wide range of Costs and Qualities. They differ a lot in their Architecture and style of tackling Problems. This is why we decided to make a Comparison of some large Open-Source tool stacks available on the Market

## Thesis Structure

In the first part the Paper describes the general aspects of Monitoring and the general Metrics. The part also discusses the characteristics of the environments and their special interfaces. After the introduction the tools will be introduced on there own. As a conclusion to this work a General overview in form of some table is given. And a conclusion to the question of the best monitoring tool is given.

**Technical Details:** In this chapter all technical aspects of the Test environments and Tools will be explained. Also the Term Stack will be illustrated.

**Collectors:** Explains all details and methods of the client based tools.

**Database:** Analyzes the different types of Methods of Storing Time related Data

**Visualizaion:** Illustrates how time related data can be presented in general and which Visual limitation to tackle

**Alerting:** Here the different Approaches and methods to inform the User/Administrator about any Problems are discussed

**InfluxData:** Company that was Founded 2012 and Provides a Full Stack Open-Source APM tool.

**Elastic:** The Elasticsearch BV provides a tool stack of APM tools. In the middle of this stack is the Elasticsearch Application which is a searching engine written in Java.

**Prometeus:** Prometheus is a Open-Source Tool stack for monitoring and alerting, with focus on reliability and simplicity.

**Zabbix:** Zabbix is an open source monitoring software for networks and applications for enterprise use

## Goals

Goal of the Study is to print out the benefits and disadvantages of the popular Open-Source Tools and Stacks for monitoring available on the Market. The work wants to Illustrate the features and technologies of the tools to make it easier for the Reader to get an overview over the different Software approaches. In particular the tools will be tested in their ability to interact with modern Cloud technologies like Docker and Kubernetes. Furthermore they will be compared by their Ability to integrate in existing environments and support of common tools and Interfaces. Moreover the Cross Compatibility of the stacks will be tested to get the best out of the tool pool.

Chapter 2

# Technical Data

## 2.1. General Stack

To explain a Stack in general, distinguish the difference between monitoring and logging. First, monitoring is running in the background and constantly collects the system data's, these data is collected by specific metrics. However logging is only triggered by a definite event or exception. With this knowledge it is a easier understanding how to establish a monitoring stack. Therefor 4 different types of tools are needed. First at all one for collect data from the system by specific metrics, the collector. Second to store, maintain and querying them, the database and last to visualize the data, the visualization tool. Often an alerting tool is also needed, but this is often integrated in the visualization tool. The logging stack has similar

## 2.2. Collector

To get Data in a centralized spot a tool is needed to collect the data were its generated and transport it to the Server or provide an Interface for the Server to collect the data. Tools for this Purpose a we call Collectors. Were are Collector for every Monitoring Purpose. Its very common that a Collector provides a general interface like an XML or JSON data or can be adapted to variable Databases to get a wide spectrum of Use-Cases. The monitored metrics is dependent on the environment and the collector also has to use over tools that provides system data to get these type of metrics. In general the data that is collected can be split up in System data and Application data. System data are all physical values like CPU load, Ram and Hard Disc Drive usage.These will be providet by cAdvisor (2.2.1) in the case of Kubernetes. Application data is dependent on the application. In the Case of monitoring Kubernetes normally the number of jobs/pods

or the number of connection per time will be monitored. These and over data will be provided by the Api-server(2.2.2) of Kubernetes.

### 2.2.1. cAdvisor

Container Advisor is tool for collection,Processing and Exporting Data of Containers. It is native Designed for Docker but can be applied to ever other container. All information about the Container is Accessible over a Rest api that gives back a JSON files with all data. A copy of cAdvisor is Deployed within every Kubernetes Pod, so every APM tool can get the metrics of the system.

### 2.2.2. Api-Server

Api-Server is a tool that provides a REST interface and is a front end for the hole Kubernetes Cluster. Over the Api-Server a user is able to interact with all Components of the cluster. The Api-Server also collects metrics witch are listed below.

- Aggreation Controler Queue: Used for Parallel Processing as an Middelware
- Registration Controller:

## 2.3. Database

The Databases for a APM are usual time-series based (2.3.1). As every other database its used to make data persistent and perform request over multiple entries to get new informations about critical values and value changes over time. Databases can offer two types of data providing methods. Most of the time the database provides a well defined interface which normals provides a authentication method to insert data into the database. Every of these Snapshot than gets a timestamp.
The over method is that the database preforms a get operation onto a interface provided by the Collector. This type of data-collection is better for static system or must be

### 2.3.1. Time-Series-Database

This is a special kind of database developed for saving time series data. This data consists of arrays which are indexed by a time stamp. By the term Time-Series also a time ranges cloud be used (as a primary key). These types of Databases can create, enumerate, update, delete and analyze time-series-data. Often they also allow you to merge multiple time-series together and make one. Like each other database, time-series-databases can also filter the data which is normally ordered ascending by time.

## 2.4. Visualization

The Visualization Tools are used to display the data stored in the databases in a nice and organized way. This is realized with plain text or by graphs. Graphs have the big advantage to be able to display the data changes over time and can very easily illustrate spikes in the data sets.Furthermore Graphs can present data in more than one way which makes it easier for humans to detect abnormal data spikes.

Usually all this information can be accessed via a web interface as this also gives a nice option for logins and distribution of permissions. This is especially useful when the data is very sensitive. Often these tools also implement easy to use Interfaces for Alerting tools, to set conditions for specific alerts, which can save a lot of time.

### 2.4.1. Graphs

As previously mentioned, the data we collected from the cluster needs to be written out of the Database and displayed in a nice and readable fashion. Thus most visualization tools use graphs to display the collected data. Using graphs not only makes the data easy to read, but it also adds the option to scale the data to our needs and preferences. This can be very useful when looking for trends in a bigger time range.
It also gives the option of color coding the data, which can be useful to either see dangerous values more quickly, or simply render multiple data streams in one graph to compare them or to see them im comparison too the hole system.

### 2.4.2. Permission Management

Most Visualization Tools have a web interface in which all the data is displayed. To make sure only authorized people can view the data, these tools usually implement

a few permission management methods. These can be ranging from simple login permissions to viewing permissions of specific data streams. Some tools allow for complete customization of the permission settings, while others offer a set of permission templates. The most popular method of authorization seems to be LDAP, as this can be used for simple and complex permission schemes alike.

#### LDAP

Written-out Lightweight Directory Access Protocol is a Network-protocol on a client-server basis. LDAP describes the communication between the client and the LDAP Directory. The data-structure of of LDAP is the so called Directory Information Tree which is organized by one suffix(root) and nodes.

### 2.4.3. Sockshop

The sockshop is a demo-application for microservices, which can be used to test monitoring microservices. It takes up quite a lot of resources, considering it is a website with a small collection of scripts behind it. This makes the sockshop a very good demo-application, as it produces enough data to test monitoring applications, or simply show the ability of a cluster to handle such tasks. Sockshop features a slim but useful HTTP-based API which allows the system owner to retrieve or post data via his own scripts or microservices to complement the application or monitor data values from the inside.

#### Architecture

Sockshop is written in multiple languages. The Front-end is written in NodeJS to give the user a nice interface. A MySQL database is used to store all the items in the shop, which is especially useful when you have complex data sets for your products. To receive the data from the MySQL database an interface written in GO is used. GO is also used for the retrieval of the users, which are stored in a Mongo database, as here no complex data structures with a lot of connections are needed. The cart of the shop uses Java to store its information inside a Mongo Database. The order process is done via a Java / .NET Core pipeline, which stores the orders inside a MongoDB. The stored orders are then processed by yet another Java pipeline to determine which orders can be shipped.

## 2.5. Alerting

To inform the developer about the system, a tool is needed which is able to sends warnings about predefined system states. The alerting tool gets one or more error codes from the controller that is normally implemented into the database or visualization tool. Some tools combine With this codes the altering tool sends a warning or error message to all people involved. Most of the tools can send over multiple platforms. The most common are: E-Mail, SMS, telegram and slack. Often tools over man then the mentioned interface and provide a api to integrate other alerting types.

Often the developers don't want to get just one alert with one message, so some of the alerting tool have the possibility to sort the alerts into groups. With this feature its possible to group cascading events that are triggered by failure. In some cases tools also supply a option to divide all alerts into critical alerts and warnings which can help the user to select the important massages.

## 2.6. Monitoring enviroment

As many Monitoring tool are flexible to the environment we chose a specific setting for the tools to monitor there Cloud skills (Kubernetes). [Voh16]

### 2.6.1. Docker

Docker is a Open-Source software that virtualize Operation Systems (OS) with the concept on containers. That means that the application an the OS is build in too one file an can be deployed out of that file with the before defined settings. This file is called image. Docker also provides a collection of pre-build images on the page https://hub.docker.com/. The software is as Kubernetes written in Go and under Apache License.

### 2.6.2. Kubernetes

Kubernetes is a Open-Source system that provides a platform for container deployment and management. It strengthen are in the field of scaling and maintaining the deployed containers. Kubernetes can run on different host on the same time. The Kubernetes Master connect all the hosts together, to form one cluster out of them. The master also

works as an interface for other systems to connect to the cluster. From the outside of the cluster its not visible on with node/nodes the application is running. This is realized by a load balancer which is also running on the Kubernetes master. Every Container in the cluster also runs a copy of cAdvisor (2.2.1) which is very important for APM, because it collects data from the system like Cpu and Ram and provides them over an interface.

### 2.6.3. GO

Go is a Programming Language that is as Kubernetes designed, implemented and updated by the Google Inc.. The basic idea about the language is to simplify the syntax compared to c and c++ by preserving it network capability and extend them in the field of Cloud technologies. The language it self is Capable of all modern concept such as Object Orientation and typification and parallelization. It was developed as an Open-Source Software in was first released in 2009.

### 2.6.4. Hardware

We were Testing all of the tools mentioned in the study on a 3 Nodes Kubernetes cluster. Every of the cluster nodes has a 4 Core Intel CPU with 2.3 GHz and 4 Megabytes of Cache . The nodes also have 8 Gigabytes of Ram each to run on. The Virtual Maschine is KVM which runes on a Fedora Linux.

Chapter 3

# Tools

---

## 3.1. Template

### 3.1.1. Toolname

Introduction Text with all infos about contributors and installation Process

Appearance

Description of the tool front-end with a Screen shot of the homepage/Dashboard

Performance

Quick analysis of the Performance (CPU/RAM usage during the test)

Interoperability

Which tools are listed to work with the tool/Which tools are test to work with the tool

Conclusion

A Quick Pro/Con of the tool and in which environment its best to use

## 3.2. Installed Tools

### 3.2.1. Searchlight (Icinga)

As in the Section Icinga 3.3.2 described, we were not able to find a pure installation of Icinga for a cluster so we tried to install Searchlight as a backup plan. Searchlight is a tool from the AppsCode Inc. (https://appscode.com/,18.12.2017) which is a company located in San Leandro Californian. Searchlight is as many other monitoring tools written in the Programming language of Go.
When first trying this over the yaml File we resived errors over the Kubernetes Cluster. There it says the tools is not able to bind the Port 8443. We could not figure out which Application is using the port but as we tried to install the app on a fresh cluster VM it turns out the yaml deployment is working fine.

#### Appearance

Icinga/Searchlight comes in a discreet blue/withe coloring. On the Dashboard which represents the homepage of the application an overview of the implemented alerts is shown. The alerts will be Colored with Green/Yellow/Red for OK/Warning/Error, so its easy to see appearing problems. The rest of the option as History and Configurations is located at the sidebar.

#### Performance

In our VM deployment the applications has allocated 160MB of RAM under Load. The Application it self is Running very smooth even on low Power Systems. As we deployed some alerts we noticed that it took about 30 seconds to validate the alert and get a first status from the system. After that pending status run Smoothly and even checks on a 30 seconds bases were no Problem for the System.

#### Interoperability

On the Github Page (https://github.com/appscode/searchlight,20.12.2017) of the Program the developer claims to be able to send Notification over Email, SMS and Chat. In the Guide there is no explicit explanation what is meant by the term Chat but the tool is cable to send Notifications to Slack(https://slack.com,03.01.2018) and Hipchat. Notification over Email can be without any third party Software send over SMTP. To

send SMS a Service like Twilio(https://www.twilio.com/,20.12.2017) is needed. On the page there is no advise that the tool is capable of interaction over an over interface then Notification.

Conclusion

Searchlight is a light weighted port of the famous monitoring tool Icinga. It runs well and has all Basic needed Features. On a deeper Look the Software revaled its weekneses. There are no advanced Possibilities to connect Searchlight with over Monitoring tool. Also there is no Graphical user interface for creating alerts, which makes it very difficult to set it up. As the tool is manly only developed and updated by 2 two people the support for new Version of Icinga and Kubernetes is limited. The fact that the company appcode has no direct correlation with Icinga is a nother counter-argument to the tool. In a nutshell the tool is good for small Kubernetes clusters that want to monitor only a few Basic metrics and have low resources.

### 3.2.2. Prometeus

The tool Prometeus is a open source Project and is hosted by the Cloud Native Computing Foundation. The project was originally built at Soundcloud. The tool can be installed on an os or on cluster like Kubernetes. Because the project is community driven, there is no official repository for Kubernetes, but with a little research we managed to find this repository (https://github.com/kayrus/prometheus-kubernetes,10.01.2018) which provides different deployments for Kubernetes.

We managed to install Prometeus so that we were able to get metrics from the Api-server and can expose them over a restful interface. Over this interface it was possible to connect Prometeus to a application outside or inside the cluster like Grafana which is recommended for Prometeus. We were not able to install the Promteus own alertmanger on the cluster. Also we were not able to get metrics from the cluster-nodes because the Prometeusversion is not cable of requesting over https.

Appearance

Prometeus provides a web ui for user to see the collected data. It is designed in a very light wighted black an with combination with some blue ascents and only shows the necessary things. The interface has no function to save the set of graphs that is selected and no option to login to keep the data save or to divide the data by some user groups.

Performance

The tool is very quickly booted and shows no Performance problems on the interface. A look on the stats shows that Prometeus is not very CPU dependent but takes up to 2GB of RAM. These high numbers of RAM is allocated over a time from about 24h.

Interoperability

It's highly recommended that Prometeus is used with a Graphing and alerting tool because its just a collector and time series database with a query engine. In our Deployment Graphana is used, which is also recommended by the Prometeus web page (https://prometheus.io/,12.01.2018). Exploiting Prometheus to any other tool is also possible as Prometeus provides the metrics that are collected in plain text

Conclusion

Prometeus is very good as a Collector for large Kubernetes Clusters with hundreds or thousands of nodes. What makes it so good is the With-Box monitoring approach, which provides way more metrics than a black-box monitoring. These fact can be used to detect problems in the system earlier and in much more detail so that the downtime can be reduced. On the over side Prometeus is not good for presenting this data and to throw alerts in case of failure.

### 3.2.3. Zabbix

[HGS15] The tool Zabbix is an open-source tool and is developed by Zabbix LLC (Limited Liability Company) since 2005 (https://www.zabbix.com,15.01.2018). Zabbix provides a all in one Solution with includes a Collector, Database, Visualization tool and a alert manager. Moreover there is a repo from monitoringartist (https://github.com/monitoringartist/kubernetes-zabbix,15.01.2018) that provides a yaml which maps Zabbix client and server on a cluster structure.

Appearance

Zabbix come with a withe and blue web interface with some Red accents. It implements a tab system with subtabs to navigate through the interface. The naming itself is not as explaining as it could be.

Performance

The general Performance of Zabbix is very good. The three parts in hole take up to 1GB of ram of the System and about 0.5 % of the Processor load. The Minimum Requirements for Zabbix are specified with Pentium 2 of 350Mhz and 256MB of Ram [MZ14].

Interoperability

Zabbix provides a different API to connect to other entities. The web interface provide a PHP script that accepts querys. Grafana itself has a extra plug in which is connectible to Zabbix and show a optimized dashboard with all relevant informations. The tool is also capable of sending alerts over email, SMS, and jabber. Zabbix gives also the ability to create customscripts and to execute them in case of a alert. For example a cli script can be triggerd over ssh.

Conclusion

Zabbix is one of the best optimized and extend open-source solutions out there. The installation was by far the simplest and overarching of all the tested tools. The configuration is complex and the naming is not at ever position as aspect-ed. By default Zabbix scales over all of the available nodes but can be scaled up manual. The best way to use Zabbix is to pare it with a Grafana instance which provides the information from Zabbix tighter and with a better overview.

### 3.2.4. ELK Stack (logstash, elasticsearch, kibana)

Elasticsearch is a tool developed by elastic.(https://www.elastic.co/, 10.01.2018) The company Elastic has many locations all around the world and seems to be very present for its customers. The ELK Stack is a logging tool which is developed java and divided up into three parts. First the logstash, the Collector of the stack. Second elasticsearch which queries the data given by the collector and last kibana the visualization tool of the stack. ELK Stack is easy deployable at the kubernetes cluster ?!?!?!. There is a complete ELK Stack kubernetes version provided by kubernetes. Just deploy each yaml and the tools are working. Hier fehlt noch die solution dass das tool richtig läuft und die erklöarung dazu.

Appearance

Kibana is the only tool in the complete Stack with a user interface. It is accessible over the homepage of the application. The site is starting with a discover page, where all logs are listed in a time chronological order. The discover page also allows to filter these logs by key words. On the leftside is the sidebar with management, devtools, visualize, timeline, discoverer and dashboard. The dashboard tab, is for creating an own dashboard with the own preferences.

Performance

Quick analysis of the Performance (CPU/RAM usage during the test)

Interoperability

Elasticsearch also offers many a tool to include alerts. The tool can send any alerts which you can query with elasticsearch. On the official website, elastics mentioned they can send the notifications on E-Mail, PagerDuty, Slack and HipChat and it also offers an interface with an webhook-output to integrate any third-party-software for messaging. With this interface it might be easy to include sending alerts with SMS like earlier mentioned the program Twilio(3.2.1).

Conclusion

A Quick Pro/Con of the tool and in which environment its best to use

## 3.3. Failed Tools

In the Process of Developing and Evaluating the APM we Discoverd a bunch of Tools that we were not able to install even that they clamed to be optimized to work on Kubernetes . In this Paragraph all the tools we wanted to include in our Report but doesn't are mentioned with a quick description of the Failure.

### 3.3.1. Graphite

Graphite is mainly for storing and Graphing data and metrics, but brings also tools that are able to collect these Metrics from the system. By the Developer it self there is no Kubernetes installation Provide but there are diverse approaches by third party members to make it runnable on a Cluster. We have tested the Repository from nanit (https://github.com/nanit/kubernetes-graphite-cluster,11.12.2017) to get Graphite running with StatsD (https://github.com/etsy/statsd.git,11.12.2017) as a metric collection tool. The Repo doesn't provide a yaml file by it self to install all the tools. The instruction leads the user to export some Variables needed for the installation. After that a deploy command is provided that pulls the docker repo and than installs it with kubectl on the Cluster. As we tried to execute this command a fail was thrown, The Node Replicas were empty, so no further commands are executable. As we were not able to install the tool on multiple Kubernetes Clusters, we installed the tool as on the Webside advertised on a Ubuntu System directly. With this installation of Graphit a time-series Data,an Monitoring and an Alerting tool is included. On the test System the Monitoring System gave values that differs from the Linux intern monitoring in values like Cpu usage or RAM. As a Solution to all this Difficulties we decided to not perform further test on the tool.

### 3.3.2. Icinga

Icinga is as ELK-Stack not a Monitoring tool. Its made for logging defined Request. The Adminestrator can decide which system to monitor an in what interval the data is pulled. Icinga it self only provides 3 system states instead of exact values. The states are OK/Warning and Error. The user decieds at which point they are triggered.
As we tried to install Icinga as we found out that there was no direct support from Incinga for Kubernetes. As our Study only describes the actual state without trying to add somthing we decidet not to try an compile Icinga into Kuberntes on oure own. Later we found out there is a third party Software called searchlight 3.2.1. Its provided by appscode on github.

Chapter 4

# Conclusion

Hier bitte einen kurzen Durchgang durch die Arbeit.

## Future Work

...und anschließend einen Ausblick

Appendix A

# LaTeX-Tipps

---

## A.1. File-Encoding und Unterstützung von Umlauten

Die Vorlage wurde 2010 auf UTF-8 umgestellt. Alle neueren Editoren sollten damit keine Schwierigkeiten haben.

## A.2. Zitate

Referenzen werden mittels `\cite[key]` gesetzt. Beispiel: [**WSPA**] oder mit Autorenangabe: **WSPA**.

Der folgende Satz demonstriert 1. die Großschreibung von Autorennamen am Satzanfang, 2. die richtige Zitation unter Verwendung von Autorennamen und der Referenz, 3. dass die Autorennamen ein Hyperlink auf das Literaturverzeichnis sind sowie 4. dass in dem Literaturverzeichnis der Namenspräfix "van der" von "Wil M. P. van der Aalst" steht. **RVvdA2016** präsentieren eine Studie über die Effektivität von Workflow-Management-Systemen.

Der folgende Satz demonstriert, dass man mittels `label` in einem Bibliopgrahie"=Eintrag den Textteil des generierten Labels überschreiben kann, aber das Jahr und die Eindeutigkeit noch von biber generiert wird. Die Apache ODE Engine [**ApacheODE**] ist eine Workflow-Maschine, die BPEL-Prozesse zuverlässig ausführt.

Wörter am besten mittels `\enquote{...}` "einschließen", dann werden die richtigen Anführungszeichen verwendet.

Beim Erstellen der Bibtex-Datei wird empfohlen darauf zu achten, dass die DOI aufgeführt wird.

**Listing A.1** lstlisting in einer Listings-Umgebung, damit das Listing durch Balken abgetrennt ist

```
<listing name="second sample">
  <content>not interesting</content>
</listing>
```

## A.3. Mathematische Formeln

Mathematische Formeln kann man $so$ setzen. `symbols-a4.pdf` (zu finden auf http://www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf) enthält eine Liste der unter LaTeX direkt verfügbaren Symbole. Z. B. $\mathbb{N}$ für die Menge der natürlichen Zahlen. Für eine vollständige Dokumentation für mathematischen Formelsatz sollte die Dokumentation zu `amsmath`, ftp://ftp.ams.org/pub/tex/doc/amsmath/ gelesen werden.

Folgende Gleichung erhält keine Nummer, da `\equation*` verwendet wurde.

$$x = y$$

Die Gleichung A.1 erhält eine Nummer:

$$x = y \tag{A.1}$$

Eine ausführliche Anleitung zum Mathematikmodus von LaTeX findet sich in http://www.ctan.org/tex-archive/help/Catalogue/entries/voss-mathmode.html.

## A.4. Quellcode

Listing A.1 zeigt, wie man Programmlistings einbindet. Mittels `\lstinputlisting` kann man den Inhalt direkt aus Dateien lesen.

Quellcode im `<listing />` ist auch möglich.

## A.5. Abbildungen

Die Figure A.1 und A.2 sind für das Verständnis dieses Dokuments wichtig. Im Anhang zeigt Figure A.4 on page xxxiii erneut die komplette Choreographie.

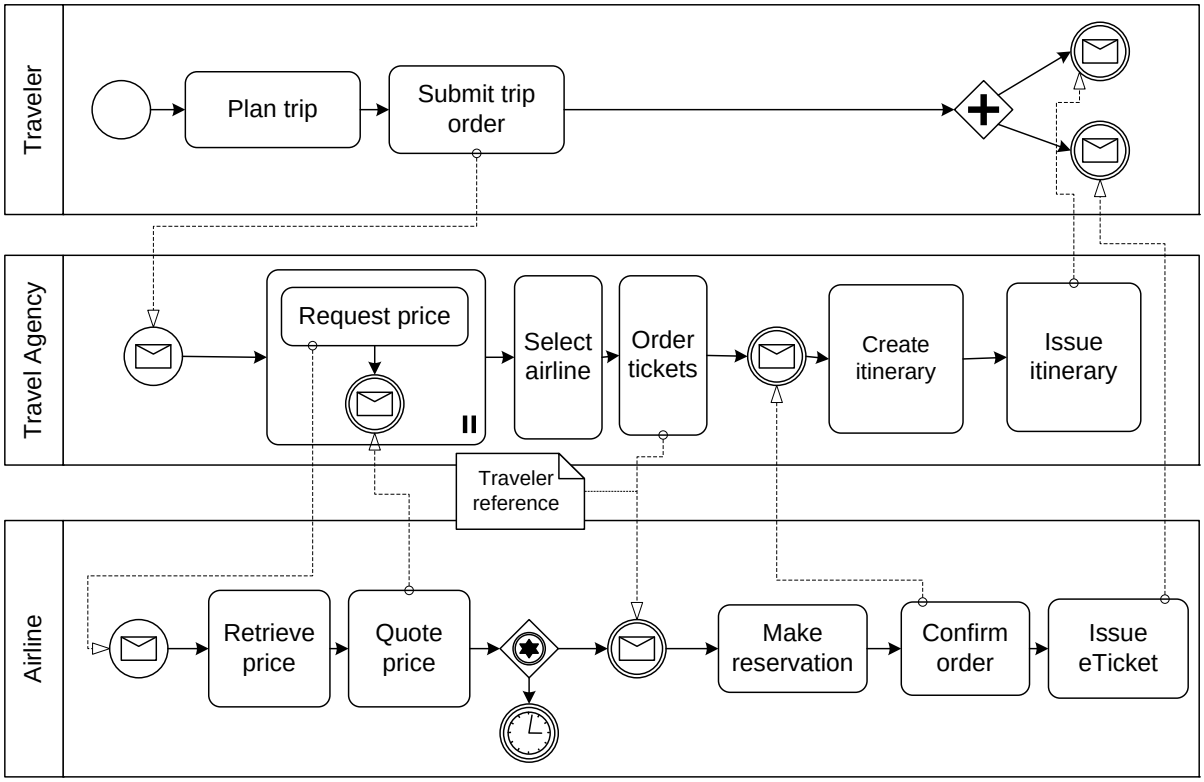



**Figure A.1.:** Beispiel-Choreographie

Das SVG in **??** ist direkt eingebunden, während der Text im SVG in **??** mittels pdflatex gesetzt ist.

Falls man die Graphiken sehen möchte, muss inkscape im PATH sein und im Tex-Quelltext `\iffalse` und `\iftrue` auskommentiert sein.

## A.6. Tabellen

Table A.1 zeigt Ergebnisse und die Table A.1 zeigt wie numerische Daten in einer Tabelle representiert werden können.

## A.7. Pseudocode

Algorithm A.1 zeigt einen Beispielalgorithmus.

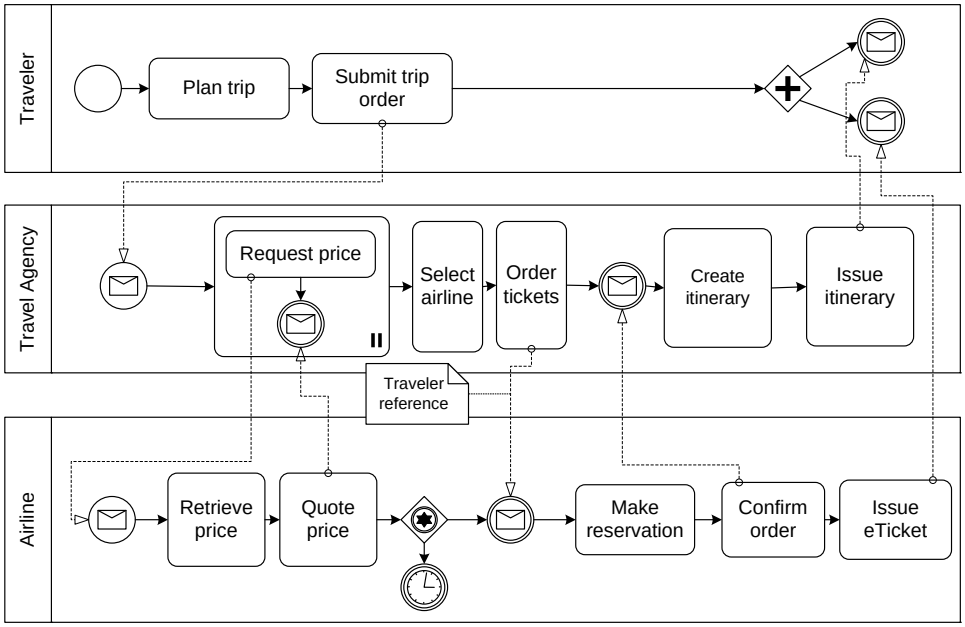
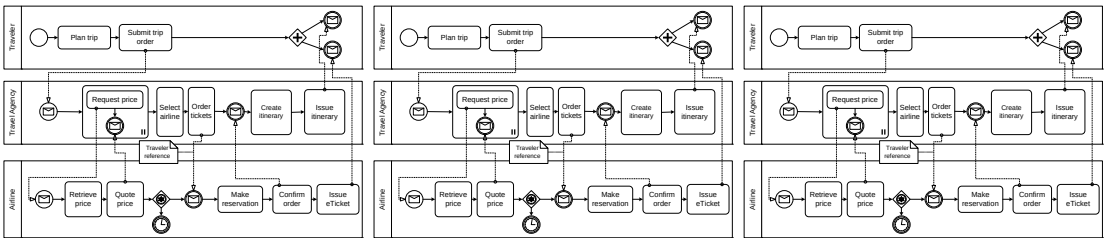



**Figure A.2.:** Die Beispiel-Choreographie. Nun etwas kleiner, damit `\textwidth` demonstriert wird. Und auch die Verwendung von alternativen Bildunterschriften für das Verzeichnis der Abbildungen. Letzteres ist allerdings nur Bedingt zu empfehlen, denn wer liest schon so viel Text unter einem Bild? Oder ist es einfach nur Stilsache?



**(a)** **(b)** **(c)** Subcaption if needed

**Figure A.3.:** Beispiel um 3 Abbildung nebeneinader zu stellen nur jedes einzeln referenzieren zu können. Abbildung A.3b ist die mittlere Abbildung.

| zusammengefasst | | Titel |
|---|---|---|
| Tabelle | wie | in |
| tabsatz.pdf | empfohlen | gesetzt |
| Beispiel | ein schönes Beispiel für die Verwendung von “multirow” | |

**Table A.1.:** Beispieltabelle – siehe http://www.ctan.org/tex-archive/info/german/tabsatz/

| | Parameter 1 | | Parameter 2 | | Parameter 3 | | Parameter 4 | |
|---|---|---|---|---|---|---|---|---|
| **Bedingungen** | **M** | **SD** | **M** | **SD** | **M** | **SD** | **M** | **SD** |
| W | 1.1 | 5.55 | 6.66 | .01 | | | | |
| X | 22.22 | 0.0 | 77.5 | .1 | | | | |
| Y | 333.3 | .1 | 11.11 | .05 | | | | |
| Z | 4444.44 | 77.77 | 14.06 | .3 | | | | |

**Table A.2.:** Beispieltabelle für 4 Bedingungen (W-Z) mit jeweils 4 Parameters mit (M und SD). Hinweiß: immer die selbe anzahl an Nachkommastellen angeben.

**Algorithmus A.1** Sample algorithm

**procedure** SAMPLE($a$,$v_e$)
  parentHandled $\leftarrow (a = \mathsf{process}) \vee \mathsf{visited}(a'), (a', c, a) \in \mathsf{HR}$
    // $(a', c'a) \in \mathsf{HR}$ denotes that $a'$ is the parent of $a$
  **if** parentHandled $\wedge$ $(\mathcal{L}_{in}(a) = \emptyset \vee \forall l \in \mathcal{L}_{in}(a) : \mathsf{visited}(l))$ **then**
    $\mathsf{visited}(a) \leftarrow$ true
    $$\mathsf{writes}_{\circ}(a, v_e) \leftarrow \begin{cases} \mathsf{joinLinks}(a, v_e) & |\mathcal{L}_{in}(a)| > 0 \\ \mathsf{writes}_{\circ}(p, v_e) & \exists p : (p, c, a) \in \mathsf{HR} \\ (\emptyset, \emptyset, \emptyset, false) & \text{otherwise} \end{cases}$$
    **if** $a \in \mathcal{A}_{basic}$ **then**
      HANDLEBASICACTIVITY($a$,$v_e$)
    **else if** $a \in \mathcal{A}_{flow}$ **then**
      HANDLEFLOW($a$,$v_e$)
    **else if** $a = \mathsf{process}$ **then**  // Directly handle the contained activity
      HANDLEACTIVITY($a'$,$v_e$), $(a, \perp, a') \in \mathsf{HR}$
      $\mathsf{writes}_{\bullet}(a) \leftarrow \mathsf{writes}_{\bullet}(a')$
    **end if**
    **for all** $l \in \mathcal{L}_{out}(a)$ **do**
      HANDLELINK($l$,$v_e$)
    **end for**
  **end if**
**end procedure**

Und wer einen Algorithmus schreiben möchte, der über mehrere Seiten geht, der kann das nur mit folgendem **üblen** Hack tun:

**Algorithmus A.2** Description

code goes here
test2

## A.8. Abkürzungen

Beim ersten Durchlaf betrug die Fehlerrate (FR) 5. Beim zweiten Durchlauf war die FR 3.

Mit `\ac{...}` können Abkürungen eingebaut werden, beim ersten aufrufen wird die lange Form eingesetzt. Beim wiederholten Verwenden von `\ac{...}` wird automatisch die kurz Form angezeigt. Außerdem wird die Abkürzung automatisch in die Abkürzungsliste eingefügt.

Definiert werden Abkürzungen in der Datei *ausarbeitung.tex* im Abschnitt '%%% acro' mithilfe von `\DeclareAcronym{...}{...}`.

Mehr infos unter: http://mirror.hmc.edu/ctan/macros/latex/contrib/acro/acro_en.pdf

## A.9. Verweise

Für weit entfernte Abschnitte ist "varioref" zu empfehlen: "Siehe Appendix A.3 on page xxvi". Das Kommando `\vref` funktioniert ähnlich wie `\cref` mit dem Unterschied, dass zusätzlich ein Verweis auf die Seite hinzugefügt wird. `vref`: "Appendix A.1 on page xxv", `cref`: "Appendix A.1", `ref`: "A.1".

Falls "varioref" Schwierigkeiten macht, dann kann man stattdessen "cref" verwenden. Dies erzeugt auch das Wort "Abschnitt" automatisch: Appendix A.3. Das geht auch für Abbildungen usw. Im Englischen bitte `\Cref{...}` (mit großen "C" am Anfang) verwenden.

## A.10. Definitionen

**Definition A.10.1 (Title)**
*Definition Text*

Definition A.10.1 zeigt ...

## A.11. Verschiedenes

KAPITÄLCHEN werden schön gesperrt...

I. Man kann auch die Nummerierung dank paralist kompakt halten
II. und auf eine andere Nummerierung umstellen

## A.12. Weitere Illustrationen

Abbildungen A.4 und A.5 zeigen zwei Choreographien, die den Sachverhalt weiter erläutern sollen. Die zweite Abbildung ist um 90 Grad gedreht, um das Paket `rotating` zu demonstrieren.

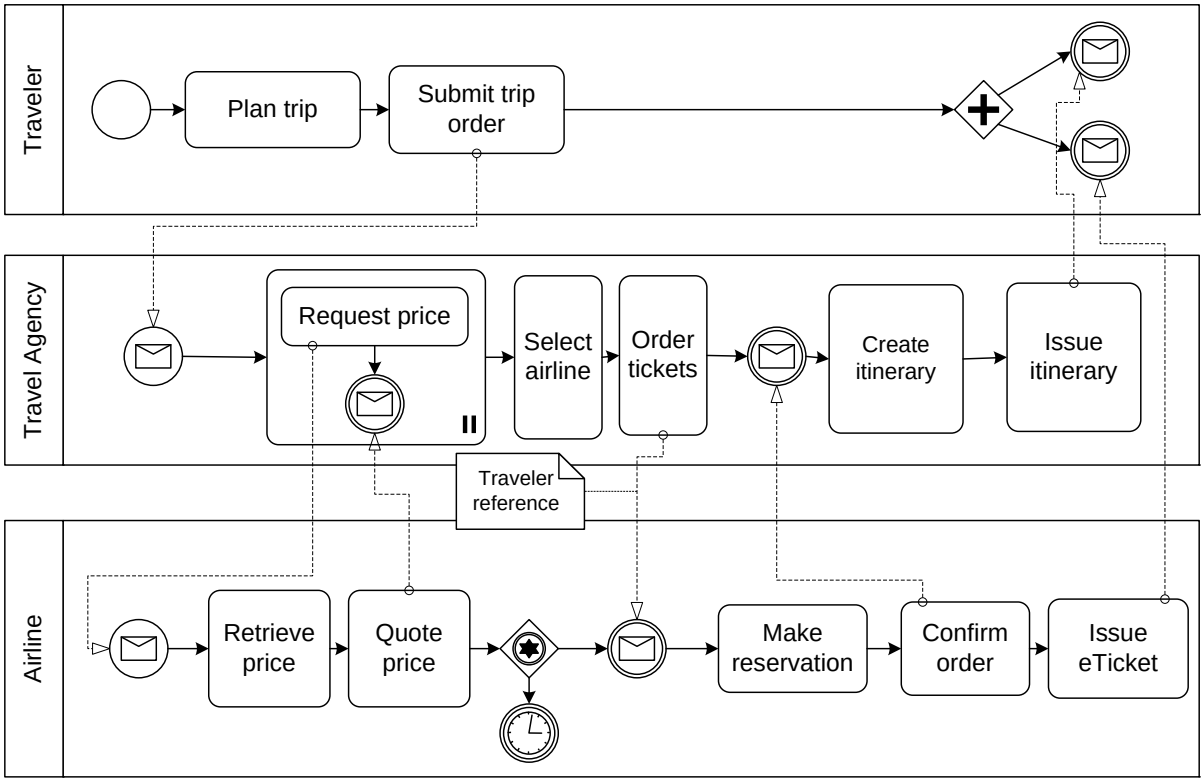



**Figure A.4.:** Beispiel-Choreographie I

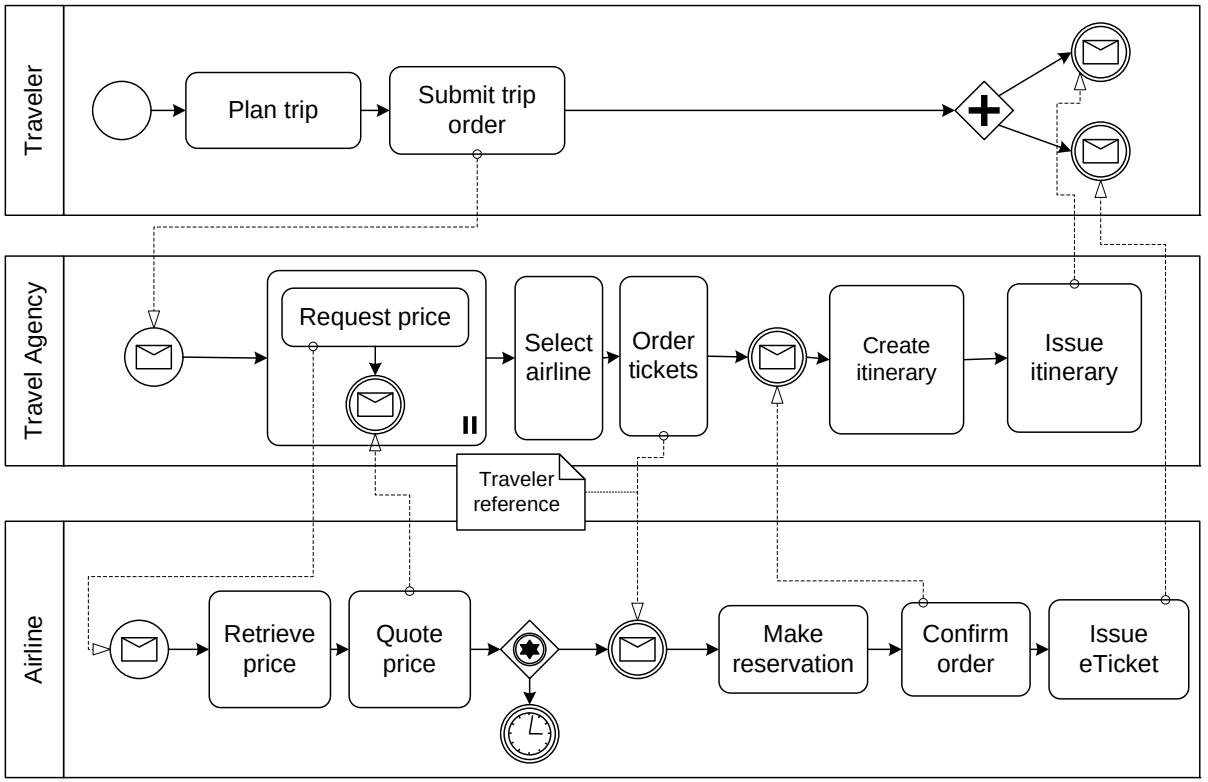



**Figure A.5.:** Beispiel-Choreographie II

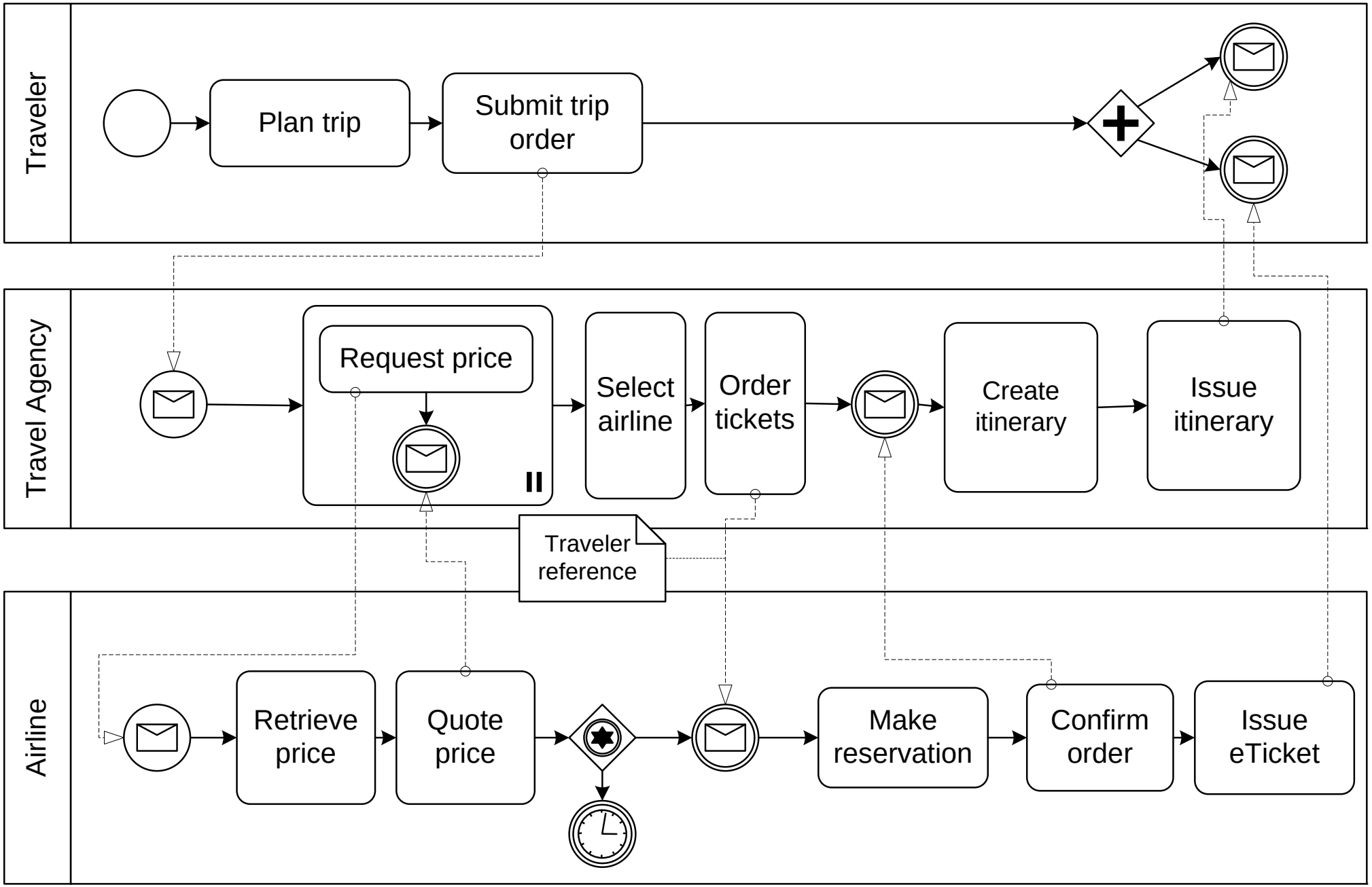



**Figure A.6.:** Beispiel-Choreographie, auf einer weißen Seite gezeigt wird und über die definierten Seitenränder herausragt

## A.13. Schlusswort

Verbesserungsvorschläge für diese Vorlage sind immer willkommen. Bitte bei github ein Ticket eintragen (https://github.com/latextemplates/uni-stuttgart-computer-science-template/issues).

Appendix A

# Bibliography

[HGS15] J. Hernantes, G. Gallardo, N. Serrano. “IT Infrastructure- Monitoring Tools.” In: *the Ieee Computer Society* (2015), pp. 88–93. ISSN: 0740-7459. DOI: 10.1109/MS.2015.96 (cit. on p. xviii).

[MZ14] O. Marik, S. Zitta. “Comparative analysis of monitoring system for data networks.” In: *2014 International Conference on Multimedia Computing and Systems (ICMCS)* (2014), pp. 563–568. DOI: 10.1109/ICMCS.2014.6911307. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6911307 (cit. on p. xix).

[Voh16] D. Vohra. *Kubernetes Microservices with Docker*. 2016. ISBN: 978-1-4842-1906-5. DOI: 10.1007/978-1-4842-1907-2. URL: http://link.springer.com/10.1007/978-1-4842-1907-2 (cit. on p. xiii).

All links were last followed on March 17, 2008.

**Declaration**

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

---

place, date, signature